Flex static code analysis

Unique rules to find Bugs, Security Hotspots, and Code Smells in your FLEX code

All rules 76 | Vulnerability 5 | Bug 9 | Security Hotspot 1 | Code Smell 61

Tags | Search by name...

Security.allowDomain(...) should only be used in a tightly focused manner
Vulnerability

The flash.system.Security.exactSettings property should never be set to false
Vulnerability

Dynamic classes should not be used
Code Smell

"LocalConnection" should be configured to narrowly specify the domains with which local connections to other Flex application are allowed
Vulnerability

"default" clauses should be first or last
Code Smell

Event types should be defined in metadata tags
Code Smell

Event names should not be hardcoded in event listeners
Code Smell

The special "star" type should not be used
Code Smell

Variables of the "Object" type should not be used
Code Smell

Methods should not be empty
Code Smell

Constant names should comply with a naming convention
Code Smell

All branches in a conditional structure should not have exactly the same implementation
Bug

Classes that extend "Event" should

"switch" statements should not be nested

Analyze your code

Code Smell | Critical | pitfall

Nested `switch` structures are difficult to understand because you can easily confuse the cases of an inner `switch` as belonging to an outer statement. Therefore nested `switch` statements should be avoided.

Specifically, you should structure your code to avoid the need for nested `switch` statements, but if you cannot, then consider moving the inner `switch` to another function.

Noncompliant Code Example

```
public function func(foo:Number, bar:Number):void
{
  switch (foo)
  {
    case 1:
      // do something
      break;
    case 2:
      switch (bar)  // Noncompliant
      {
        case 89:  // It's easy to lose sight of what's b
          // ...
          break;
        case 90:
          // ...
          break;
      }
      break;
    case 3:
      // do something
      break;
    default:
      break;
  }
}
```

Compliant Solution

```
public function func(foo:Number, bar:Number):void
{
  switch (foo)
  {
    case 1:
      // ...
      break;
    case 2:
      handleBar(bar);
      break;
    case 3:
      // ...
      break;
    default:
      break;
  }
```

override "Event.clone()"
Bug

Constructors should not dispatch events

Bug

"ManagedEvents" tags should have companion "Event" tags

Bug

Objects should not be instantiated inside a loop

Code Smell

Two branches in a conditional structure should not have exactly the same implementation

Code Smell

Constructor bodies should be as lightweight as possible

Code Smell

Only "while", "do" and "for" statements should be labelled

Code Smell

Statements, operators and keywords specific to ActionScript 2 should not be used

Code Smell

"for" loop stop conditions should be invariant

Code Smell

Unused function parameters should be removed

Code Smell

```
}

public function handleBar(bar:Number):void
{
  switch (bar)
  {
    case 89:
      // ...
      break;
    case 90:
      // ...
      break;
  }
}
```

Available In:

sonarcloud sonarqube